IN THE CLAIMS

Please amend claim 72 and 83 as indicated below.

1–10. (Cancelled).

11. (Previously Presented) A microprocessor based system comprising:

a memory controller coupled to a memory and to a plurality of microprocessors, each microprocessor having a bus interface that implements a plurality of bus phases for a memory transaction; and

a global arbiter, coupled to each bus interface and to said memory controller, wherein said global arbiter is configured to:

receive one or more requests for a memory transaction;

assign a global order to each of the one or more requests;

execute according to the global order, a snoop corresponding to each request, wherein each snoop comprises determining whether one or more of the plurality of microprocessors has data associated with a corresponding request; and

respond to each request according to the global order.

12. (Previously Presented) The microprocessor based system as recited in claim 11 wherein said plurality of bus phases implemented by said bus interfaces comprise:

a request phase, where memory requests are produced by said bus interfaces;

a snoop phase, where a determination is made whether a microprocessor of the plurality of microprocessors has data associated with a memory request; and

a response phase, where data associated with a request is received.

13. (Original) The microprocessor based system as recited in claim 12 wherein said memory requests comprise:

a request to share;

a request to own; and

a write back.

14. (Original) The microprocessor based system as recited in claim 13 wherein said request to share results from a read miss in a cache.

15. (Original) The microprocessor based system as recited in claim 13 wherein said request to own results from a write miss in a cache.

16. (Original) The microprocessor based system as recited in claim 13 wherein said write back results from a snoop to a modified cache line.

17. (Previously Presented) The microprocessor based system as recited in claim 11 wherein said determining comprises communicating a query corresponding to a request to each of the plurality of microprocessors.

18. (Previously Presented) The microprocessor based system as recited in claim 11 wherein a first microprocessor's bus interface couples said first microprocessor to a first bus.

19. (Previously Presented) The microprocessor based system as recited in claim 18 wherein a second microprocessor's bus interface couples said second microprocessor to a second bus.

20. (Original) The microprocessor based system as recited in claim 19 wherein said first bus and said second bus are the same.

21. (Original) The microprocessor based system as recited in claim 19 wherein said first bus and said second bus are different.

22. (Original) The microprocessor based system as recited in claim 19 wherein said first bus and said second bus have different timing latencies.

23. (Original) The microprocessor based system as recited in claim 11 wherein said memory controller comprises:

a bus interface for coupling to said global arbiter.

24. (Previously Presented) The microprocessor based system as recited in claim 11 wherein said global arbiter comprises a bus interface for coupling to each of said plurality of microprocessors and said memory controller.

25. (Previously Presented) The microprocessor based system as recited in claim 11 wherein said determining comprises:

communicating a query corresponding to a given request to each of the plurality of microprocessors; and

waiting until a response to the query corresponding to the given request is received from each of the plurality of microprocessors before responding to the given request.

26. (Original) The microprocessor based system as recited in claim 11 wherein said global arbiter comprises:

a plurality of request queues;

a plurality of snoop queues; and

a plurality of response queues;

wherein each of said queues stores a plurality of requests, snoops, and responses, respectively.

27. (Previously Presented) The microprocessor based system as recited in

claim 11 wherein said global arbiter comprises:

ordering and arbitration logic for receiving a plurality of requests from said plurality of microprocessors, and for ordering said plurality of requests into a global order.

28. (Original) The microprocessor based system as recited in claim 27 wherein said ordering and arbitration logic initiates snoops according to said global order.

29. (Previously Presented) The microprocessor based system as recited in claim 28 wherein said snoops have differing latencies depending on bus characteristics of busses coupling said global arbiter to said plurality of microprocessors.

30. – 42. (Cancelled)

43. (Previously Presented) A multiphase protocol for insuring coherency between agents that share a memory through disparate fabrics, the protocol comprising:

a request phase, where memory requests are presented to a global arbiter, said global arbiter ordering said memory requests into a global order;

a snoop phase, during which the global arbiter executes a snoop corresponding to each request according to said global order, wherein a snoop determines whether one or more agents coupled to the disparate fabrics have data associated with a corresponding memory request; and

a response phase, where responses to said memory requests are provided according to said global order upon completion of their associated snoops.

44. (Original) The multiphase protocol as recited in claim 43 wherein the agents comprise a plurality of processor cores.

45. (Original) The multiphase protocol as recited in claim 44 wherein each of the agents further comprise a cache.

46. (Original) The multiphase protocol as recited in claim 43 wherein the disparate fabrics comprise a plurality of interfaces.

47. (Original) The multiphase protocol as recited in claim 46 wherein at least one of said plurality of interfaces comprises a bus.

48. (Original) The multiphase protocol as recited in claim 43 wherein at least one of said plurality of interfaces comprises a serial fabric.

49. (Original) The multiphase protocol as recited in claim 48 wherein said serial fabric comprises PCI-Express.

50. (Original) The multiphase protocol as recited in claim 43 wherein said request phase, said snoop phase, and said response phase pertains to each of said memory requests.

51. (Original) The multiphase protocol as recited in claim 50 wherein said memory requests may overlap with said request, snoop, and response phases for another one of said memory requests.

52. (Original) The multiphase protocol as recited in claim 43 wherein said memory requests comprise:

memory reads; and

memory writes.

53. (Previously Presented) A method for providing latency independent coherence among a plurality of agents that share a memory, the method comprising:

establishing three phases for memory requests comprising:

a request phase;

a snoop phase; and

a response phase;

a global arbiter receiving one or more requests for a memory transaction;

when multiple memory requests are outstanding, the global arbiter establishing a global order for the requests, each of the requests submitted during the request phase;

entering a snoop phase for each of the requests, following its request phase, the snoop phase comprising the global arbiter querying, according to the global order, the plurality of agents to determine whether they contain data pertaining to the requests; and

entering a response phase for each request after the plurality of agents have responded to the snoop phase for the request, the response phase providing data pertaining to the request to its requesting agent.

54. (Previously Presented) The method as recited in claim 53 wherein the request phase comprises submitting a request from an agent to the global arbiter.

55. (Original) The method as recited in claim 54 wherein the request phase further comprises receiving the request by the global arbiter, and ordering the request according to the global order.

56. (Previously Presented) The method as recited in claim 53 wherein the

snoop phase comprises the global arbiter communicating queries corresponding to the requests to the agents that share the memory according to the global order.

57. (Previously Presented) The method as recited in claim 56 wherein the snoop phase further comprises the global arbiter awaiting a snoop response from the agents before proceeding to the response phase.

58. (Original) The method as recited in claim 53 wherein the response phase comprises providing data associated with a request to the agent that generated the request, according to the global order.

59. (Original) The method as recited in claim 53 wherein the global order causes response phases associated with each request to be completed, in order, without regard to latencies between memory requests.

60. (Original) The method as recited in claim 53 wherein the global order causes response phases associated with each request to be completed, in order, without regard to latencies between a global arbiter and its agents.

61. (Previously Presented) A computer readable storage medium containing instructions that, when executed by a processor, enable the processor to provide:

a global arbiter, coupled to each of a plurality of processor cores;

a bus interface for each of the processor cores to couple the processor cores to the global arbiter; and

a bus interface to couple the global arbiter to a memory;

wherein said global arbiter receives memory requests from each of the processor cores, establishes a global order for the requests, and initiates a snoop phase for each of the requests according to the

global order, each snoop phase comprising the global arbiter querying the plurality of processor cores to determine whether they contain data pertaining to the memory requests; and

wherein the processor cores respond to the snoop phase initiated by the global arbiter at different times.

62. (Previously Presented) The computer readable storage medium as recited in claim 61 wherein the global arbiter comprises:

request logic, for receiving requests from the processor cores;

snoop logic, for communicating queries associated with received requests from the global arbiter to the processor cores; and

ordering logic, for establishing a global order for received requests, and for causing said snoop logic to communicate the received requests to the processor cores according to the global order.

63. (Previously Presented) The computer readable storage medium as recited in claim 62 wherein the global arbiter further comprises:

response logic, for causing responses to the received requests to be provided upon completion of their associated snoop.

64. (Previously Presented) The computer readable storage medium as recited in claim 62 wherein the global arbiter further comprises:

response logic, for causing responses to the received requests to be provided to the processor cores according to the global order.

65. – 69. (Cancelled)

70. (Previously Presented) The multiphase protocol of claim 43, wherein a snoop further comprises the global arbiter communicating a query corresponding to a memory request from a first agent to all other agents that share the memory.

71. (Previously Presented) The multiphase protocol of claim 43, wherein in determining whether another agent has data associated with a given memory request, the global arbiter is further configured to:

communicate a query corresponding to the given memory request from a first agent to all other agents that share the memory; and

wait until a response to the query is received from each agent before responding to the given memory request.

72. (Currently Amended) A memory controller coupled to a memory and to a plurality of agents configured to share the memory, wherein the memory controller is configured to:

receive one or more memory requests;

assign a global order to each of the one or more requests;

execute according to the global order, a snoop corresponding to each request, wherein each snoop comprises determining whether one or more of the plurality of agents has data associated with a corresponding request; and

respond to each request according to the global order;

ordering logic for establishing a global order for said requests, and for insuring ~~that~~ initiation of said ~~initiated~~ snoops conform to said global order.

73. (Cancelled).

74. (Previously Presented) The memory controller of claim 72, wherein in determining whether another agent has data associated with a first request from a first agent, the memory controller is further configured to communicate a query corresponding to the first request to all other agents of the plurality of agents.

75. (Previously Presented) The memory controller of claim 72, wherein in determining whether another agent has data associated with a given request, the memory controller is further configured to:

communicate a query corresponding to the given request to all other agents of the plurality of agents; and

wait until a response to the query is received from each agent before responding to the given request.

76. (Previously Presented) The memory controller of claim 72, wherein the plurality of agents comprise microprocessor cores and I/O devices.

77. (Previously Presented) The memory controller of claim 76, wherein each of said microprocessor cores includes a cache.

78. (Previously Presented) The memory controller of claim 72, wherein said plurality of agents and said memory controller are coupled to a common bus.

79. (Previously Presented) The memory controller of claim 72, wherein said plurality of agents and said memory controller are coupled to disparate buses.

80. (Previously Presented) The memory controller of claim 72, further configured to provide coherency between caches within the plurality of agents and the memory.

81. (Previously Presented) The memory controller of claim 72, wherein each agent monitors requests to determine whether it holds data associated with a request from another one of the plurality of agents.

82. (Previously Presented) The memory controller of claim 72, wherein if a first agent determines that another one of the plurality of agents requests data that is within the first agent, the first agent informs the memory controller.

83. (Currently Amended) The memory controller of claim 72, further comprising:

request logic, for receiving requests from each of the plurality of agents;

snoop logic, for initiating said snoops to the plurality of agents; and

wherein said initiated snoops are latency independent with respect to said requests.

84. (Previously Presented) The memory controller as recited in claim 83, wherein said request logic comprises a plurality of queues for receiving said requests from the plurality of agents and for presenting said requests to said ordering logic.

85. (Previously Presented) The memory controller as recited in claim 83, wherein said snoop logic comprises a plurality of queues for storing snoops conforming to said global order.

86. (Previously Presented) The memory controller as recited in claim 83 wherein said ordering logic causes said snoop logic to initiate snoops according to said global order.

87. (Previously Presented) The memory controller as recited in claim 83 wherein said global order is established according to a first-in first-out rule.

88. (Previously Presented) The memory controller as recited in claim 83 wherein said ordering logic insures that responses to said requests are performed according to said global order.

89. (Previously Presented) The memory controller as recited in claim 83 further comprising:

response logic for insuring that responses to said requests are performed according to said global order.

90. (Previously Presented) The memory controller as recited in claim 83 further

comprising:

a plurality of bus interfaces for coupling said memory controller to the plurality of agents.

91. (Previously Presented) The memory controller as recited in claim 90 wherein at least one of said plurality of bus interfaces couples to a bus that is different than the others.

92. (Previously Presented) The memory controller as recited in claim 90 wherein at least one of said plurality of bus interfaces couples to a bus that supports a plurality of virtual channels.

93. (Previously Presented) The memory controller as recited in claim 92 wherein said at least one of said plurality of bus interfaces comprises a bus interface to PCI-Express.